The Computer Journal Advance Access published December 21, 2007

doi:10.1093/comjnl/bxm103

A Policy-Driven Service Composition Method for Adaptation in Pervasive Computing Environment

Baopeng Zhang*, Yuanchun Shi and Xin Xiao

Key Laboratory of Pervasive Computing, Ministry of Education Department of CS, Tsinghua University, Beijing, P.R. China
**Corresponding author: zbp02@mails.tsinghua.edu.cn*

Service composition allows distributed application, such as multimedia application, to be composed from atomic service units and to adapt dynamically to users' requirements and environment conditions in pervasive computing system. It augments the adaptation action space for the application of pervasive computing. According to the multidimensional QoS (Quality of Service) requirement of pervasive computing system, we proposed a comprehensive service composition method to enhance the capability of application adaptation. First, according to a hierarchy policy model and a policy specification language, strengthened by event calculus, service discovery policy action integrating the situation of user, application, environment and resource can be triggered. Secondly, the proposed physical space model can support the location-aware service discovery and explicit range query to improve the efficiency of the query. To the end, an adaptation policy evaluation model is utilized to maximize an evaluation criterion–quality of satisfaction of users and environment by optimizing the optional service selection and the composition path. Through experiment and discussion of the algorithm, the paper further illustrates the great potential advantage of the solution to service composition.

Keywords: policy-driven method; pervasive computing; service discovery; adapation; service composition

Received 23 February 2007; revised 16 October 2007

1. INTRODUCTION

In the last few years, with the increasing dependency on the wired/wireless communication and the distributed application services, the next-generation networks are envisioned to support the dynamic provision of network service, evolved into an indispensable service delivery infrastructure and catered for the specific Quality of service (QoS) requirements of users and applications. Multimedia service provision can be dynamically created by a compositional approach using the distributed service components, which are named application services. Furthermore, newly emerging pervasive computing environments such as smart space [1] provide the many multimodal interface services to improve the user's experience. So an important requirement for pervasive computing system is the ability to adapt itself at runtime to deal with such situations as resource variability, user requirement, environment condition, and user mobility. Therefore, three kinds of strategies—reducing the user-perceived quality, reserving resources to guarantee the quality of service and suggesting an appropriate action to users—efficiently enlarge the adaptation space of pervasive computing systems [2]. In this paper, an application service refers to a self-contained software unit that provides specific application function, such as various multimedia transcoder, image scaling, text-to-speech services and so on. The application can be regarded as a composite service formed by diverse application service in different time or space pattern. Also, an application and an application service are referred to as a task and a subtask, respectively. We use two kinds of expressions by turns in this paper. Similar to the abstract of a Task Graph in parallel computing, most tasks and subtasks that we are concerned about involve several specialized heterogeneous devices and application services that communicate with each other. Therefore, this kind of task graph formulation is more general than the one used in the parallel computing [3]. How to set up the application using the application services in wired/wireless network is an important issue which is termed service composition. In the presence of resource (e.g. networks, device etc.) fluctuation, user preference and mobility, how to meet user and application requirements by handling application services to interoperate spontaneously is termed service adaptation. In our view, solution to the composition and adaptation problem should be QoS-aware and Semantic-aware, in that composition should satisfy QoS requirement of the application based on the end-to-end QoS information, available resources of computing environment and physical space bound constraints [4].

This paper proposes a new policy-based approach to handle the issues of service adaptation and service composition decision in pervasive computing environment. The proposed work is inspired from automated policy-based management framework for differentiated communication systems [5]. We proposed a hierarchical policy model to facilitate the mapping among higher level requirements of user, application and environment in order to utilize the service discovery to satisfy the adaptation goal. According to the network-level constraints, environment constraints and their goals, we trigger appropriate adaptation action for multimedia application, provide the algorithm to assess the rationality of service selection and perform the optimization of the adaptation action to be taken. The remainder of this paper is organized as follows. Section 2 introduces related work. Section 3 gives the service composition and adaptation problem definition. Section 4 discusses proposed policy hierarchy model, policy description and policy-driven service discovery scheme. Section 5 gives the corresponding evaluation of algorithm of adaptation policy and discussion about solution to algorithm-relative service composition problem. In Section 6, the paper ends with conclusion and future work.

2. RELATED WORK

Recently, several research works have addressed the service composition and adaptation issue. Nancy Samaan presented a novel framework for adaptive policy-based QoS management in wired/wireless network [5]. New policies, the function of different parameters such as time, location and cost are assembled at runtime to control network behavior with the change of the surrounding environment conditions.

The SpiderNet service composition approach provides multi-constrained statistical QoS assurances for composed distributed multimedia services, and its service adaptation approach uses an ordered coordination algorithm to automatically generate the service adaptation plan [6]. But this method does not take account of the user requirements.

As a QoS-aware middleware supporting quality-driven web service compositions, Agflow proposes a service quality model of web service, and two kinds of service selection approach — local optimization and global planning [7].

Our early works [8] proposed a novel algorithm known as Adaptive Multimedia Transport Model (AMTM). It implemented an adaptive supporting platform for multimedia delivery, which can dynamically transcode the multimedia data to accustom them to the network variation.

In this paper, we use a similar policy-driven approach and service composition application mode to deal with adaptation problem, but our problem space has much more high-level user requirement and space constraints coming from the pervasive computing environment.

3. PROBLEM DEFINITION

In this section, a pervasive computing system entity model is presented and some basic concepts and features of service composition and adaptation factor in pervasive computing environment are explained. We formally define the composition-based adaptation problem.

3.1. Entity relation model

As an open distributed system, pervasive computing paradigm contains some self-configuration entity objects: *Environment*, *User*, *Task*, *AtomicTask* and *Service*. We formalize the logic relation among them to make the factors of influencing service composition and adaptation explicit.

Ontology is generally defined as the representation of a shared conceptualization of a particular domain. We represent the abstract of entity relation by means of an ontology. The entities termed class defines a group of individual that share some attributes. The abstract of relation is denoted as property, which defines the connection among entities. As shown in Fig. 1, we define some property primitives as:

- Control Construct, which denotes time or space pattern of application composed by application services. It defines the semantic logic relation of composite service.
- InterpretedAs, which denotes application services map to the concrete service class instances.

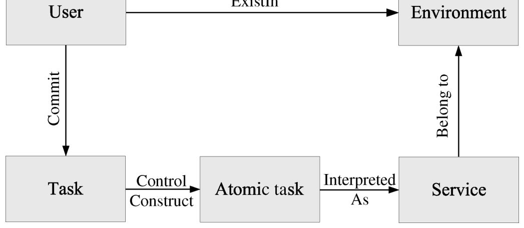

FIGURE 1. Pervasive computing system entity relation.

- BelongTo, which denotes environment contains services. The use of service class instances should meet the environment rule and goal.
- ExistIn, which denotes user location in physical space or social community.
- Commit, which denotes users' requirement for Task and AtomicTask.

3.2. Service composition graph

We describe a task using a Directed Acyclic Graph (DAG) called SCG (λ).

DEFINITION 1. *SCG(λ) is defined as* $\lambda = \langle S, L_c, L_d \rangle$

$$S = \{s_k \mid 1 \le k \le \mid S|\},$$
$$L_c = \{cl_k \mid cl_k = s_i \longleftrightarrow s_j, 1 \le k \le \mid L_c|\},$$
$$L_d = \{l_k \mid l_k = s_i \longrightarrow s_j, 1 \le k \le |L_d \mid\}.$$

This service composition graph (SCG) can be constructed by three kinds of basic graph elements. In Fig. 2, the circle nodes represent the overlay network node containing the special service; cl_k represents path containing both control flow and data flow; l_k represents data flow path only. We define that the control path does not shift unless the user allows. As we know, some multimedia contents have independent media objects, e.g. photo, audio and video flow which can have different paths. Content adaptation service may be linked in SCG representing distributed application. Some composition service models are proposed, such as directed acyclic graph [9] and ConcurTaskTrees [10], to express the relation among the atomic tasks.

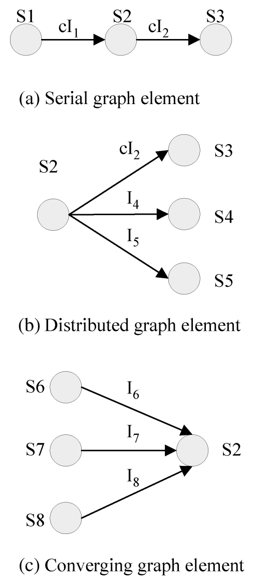

FIGURE 2. Basic service graph element.

3.3. Multidimensional QoS requirement

Traditionally, given the heterogeneity of application, user preferences, underlying operating systems, network and devices, a layered QoS model including user, application and resource layer is presented [11]. Most research works aim to translate non-functional parameters of QoS at each layer to network objectives. But, in our paradigm, it is not enough. The requirements of functional QoS parameters and Environmental QoSs bring new challenges.

As shown in Fig. 3, the user layer describes user preference and requirement such as the requisite service class SCU, perceptive media type and quality PU, time delay DU etc. U_i ⟨SCU, PU, DU, . . .⟩.

The application layer describes composite service semantic logic SL, Execute path EP, end to end delay D_g etc. A_j ⟨SL, EP, D_g, . . .⟩.

The environment layer describes physical space model PSM, service sets SS, maximal user number N_{MAX}, current user number N_{cur}, total throughout T etc. E_k ⟨PSM, SS, N_{MAX}, N_{cur}, T, . . .⟩.

The resource Layer describes NET (bandwidth BW, propagation delay D_p, etc.), service quality SQ (function, location, execution duration, successful execution rate, adaptability, availability, bandwidth requirement etc.), device resource DR (presentation quality P_d, CPU, battery power etc.) etc. R⟨NET, SQ, DR, . . .⟩.

As mentioned above, every layer of QoS has some semantic-related functional description and non-functional parameters. According to this multi-dimensional model, context-aware service composition adaptation problem in pervasive computing environment can be expressed as:

Adaptation Problem:
For application A_i, *Adaptation* (A_i) iff
Compatiblewith (A_i, *U, E, R*) &
Satisfiedwith (*U, E*) & *Consistency* (A_i, A_i·SL).

DEFINITION 2. *Compatiblewith* (A_i, *U, E, R*)

It includes requirement compatibility and policy compatibility. Requirement compatibility claims that the user requirement for service should satisfy the capability of resources available. We define that requirement compatibility is valid

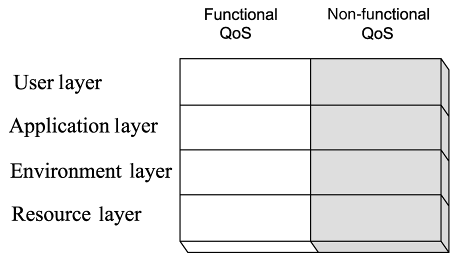

FIGURE 3. Multidimensional QoS model.

if and only if:

$$U_i.\mathrm{PU} \preceq R_i.\mathrm{DR}.P_d,\ U_i.\mathrm{SCU} \preceq R_i.\mathrm{SQ}$$

and given adaptation policy set of user, application, environment and resource

$$\Gamma_U = \{P_{Ur1}, \ldots, P_{Urm}\}, \Gamma_A = \{P_{Ar1}, \ldots, P_{Arn}\},$$
$$\Gamma_E = \{P_{Er1}, \ldots, P_{Erl}\}, \Gamma_R = \{P_{Rr1}, \ldots, P_{Rrd}\},$$

then the policy compatibility, as a kind of meta-policy, expresses priority of several policy set in order to avoid policy conflict. It is defined as:

$$\forall i,\ 1 \le i \le n\, P_{Ari} \in \Gamma_A \text{ is valid iff } \exists P_{Ur}, P_{Er},$$
$$P_{Rr}, P_{Ur} \preceq P_{Rr} \preceq P_{Er}.$$

Definition 3. *Satisfiedwith (U, E) U* $\Leftrightarrow$ *E*

$$\forall A_i \in U_j,\ U_j.\ \mathit{Location} \subseteq E,\ U_j \Leftrightarrow E \text{ iff}$$
$$U_j.\mathrm{SCU} \subseteq E.\mathrm{SS}\ \&$$
$$E.\mathrm{PSM}(U_j, U_j.\mathrm{SCU}.\mathit{Location}) \le w$$

w is a range value of location distance defined by user.
This definition represents the relation between user requirement and environmental characteristics. It emphasizes the service selection must satisfy the location range and service availability of physical environment.

Definition 4. *Given two service components,*

$$\forall \mathrm{SC}_a, \mathrm{SC}_b \in A_i,\ \mathit{Consistency}\,(A_i, A_i.\mathrm{SL}):$$
$$A_i \Rightarrow A_i.\mathrm{SL} \text{ iff}$$

- Data-type consistency

$$A_i.\mathrm{SC}_a.\mathit{DataType}\,(\mathit{output}) \subseteq$$
$$A_i.\mathrm{SC}_b.\mathit{DataType}\,(\mathit{input})$$

- QoS requirement consistency

$$A_i.\mathrm{SC}_a.\mathrm{QoS}\,(\mathit{output}) \preceq A_i.\mathrm{SC}_b.\mathrm{QoS}\,(\mathit{input})$$

- Semantic concept consistency
  Given concept class of service s is expressed as *conceptof*(s) that represents the abstract ideas and action, substitute service component SC_s of service SC_b satisfies

$$\mathit{conceptof}\,(\mathrm{SC}_s) \preceq \mathit{conceptof}\,(A_i.\mathrm{SC}_b)$$

The three consistency claims show the indispensable relation between services for composition. In the process of service composition, the input and output character of two services should match each other. As an adaptation mechanism, service substitution must answer for application semantic, i.e. expressing same semantic content by different modalities.

4. POLICY HIERARCHY MODEL

Our work follows the same concept of adaptation [12], the use of policies at different layers allows user, application and environment to dynamically specify their requirements in terms of high level goals. We use the policies to trigger the service adaptation and adjust the service composition process for better performance. We adopt a hierarchy model to capture user and environment requirement more efficiently, weigh the resource constraints and optimize the action of service composition and adaptation. This is shown in Fig. 4.

Based on the multidimensional QoS model mentioned before, we construct the policy hierarchy model. Earlier works that addressed the issue of policy adaptation can be classified as three categories. The first category of schemes dynamically change different parameters of a QoS policy to specify new attributes values. The second category performs adaptation by enabling/disabling a policy from a set of predefined QoS policies [13]. The last category tackles this issue via posing it as a problem of learning process to assemble new policies at runtime. The most important challenge of assembling policies of every layer in service composition and adaptation problem lies in choosing the appropriate policy actions by assessing the effect of actions. Two issues should be taken into account. The first issue is local optimization for service selection by considering the user and environment requirement. The second issue is global optimization for guaranteeing the end-to-end QoS. As a result, we define policy-driven adaptation process as two parts. The first one is confirming the policy validity which complies with the current context information. The second one is selecting the optimal policy parameters and action according to the multi-level entity requirement.

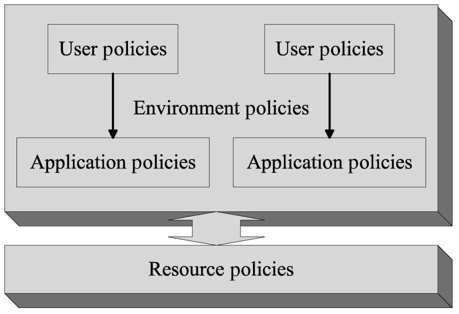

FIGURE 4. Hierarchical policy model.

4.1. Policy description

Highly dynamic characteristic of pervasive computing environment makes the interaction among the entity more complex and hard to control. Policies are utilized to guide the behavior of entities in security, management and even network domain. Every entity can define the respective policy. We believe that the policy-driven method can be a flexible approach to deal with the interoperation of entities. In order to enable entities in pervasive computing systems to understand and interpret their behavior correctly, policies are described with a semantic-rich language that allows entities to have a common understanding of policies, roles and other attributes. Towards the end, many policy languages are proposed, such as Ponder [14], WSPL [15] and Rei [16]. Some kinds of normative notion should be supported by the policy language and strategy [16]. The deontic logic notion represents the logical relationships among propositions that assert certain actions or states of affairs. Three basic deontic logic notions that we focus on are:

- **Right.** The permission is defined for the entity to execute a specified action in the particular context. A user playing a specific role can use specified service in the pervasive computing environment.
- **Obligation.** The entity must assume responsibility that action should be performed when specified conditions are satisfied.
- **Prohibition.** The negative authorizations that the actions cannot be performed by an entity.

The speech act is an act that a speaker performs when making an utterance. In the distributed computing environment, it is a communicative policy description. We also use some speech act to enforce the decentralized control between the entities:

- **Delegation.** An entity can give a right to another entity. The delegated entity can perform the specified action according to local constraint conditions. For example, service entity can delegate the service discovery or service composition action to other service entity for distributed session control.
- **Request.** An entity can claim the right from other entity. This is the inverse act of delegation.
- **Cancel.** This is the nullifying act for cancel the request for right or delegation.

Due to a number of possible changes of the context can influence distributed application execution. we use policy to express the behavior of entities in the pervasive computing environment based on the respective rule. So context changes can be dealt with in static or dynamic conditions.

For the sake of the distributed control, it is necessary for policy to introduce event operation. Just like a simple but expressive language, PDL can be described as a real-time specialized production rule system to define policies by using the event-condition-action rule paradigm of active database [17]. PDL consists of three basic classes of symbols: primitive event symbols, action symbols and function symbols. Policies are described by a collection of propositions of two types: policy rule propositions and policy defined event propositions that are expressed as follows:

Events *cause* **Action** if **Condition**

Events *trigger* **PDE** if **Condition**

We believe that the event is a kind of time-point information, but it cannot express the time-varying properties. The high-level context information is primarily such time-varying properties, and their expression can make adaptation mechanism more accurate and efficient. We define the events that characterize time-varying properties as situation information, and in the pervasive computing environment, the situation information has important meanings for the adaptation operation. So we introduce the event calculus to enhance the capability of policy rule expression. The event calculus was originally introduced by Bob Kowalski and Marek Sergot as a logic programming framework for representing and reasoning about actions and their effects [18]. Its ontology consists of (i) a set of time points, (ii) a set of time-varying properties called fluents, and (iii) a set of event types or actions. The logic includes four base predicates, two auxiliary predicates and four domain-independent component predicates. This is formalised as below:

Base predicates:

Happen (a, t) : event a (or action) occurs at time point t.

Initiates (a, f, t) : if event a were to occur at t, it would cause fluent f to be true.

Terminates (a, f, t) : if event a were to occur at t, it would cause fluent f to be false.

HoldsAt (f, t) : fluent f is true at t

Auxiliary predicates:

$$\begin{aligned} Clipped\,(t_1, f, t_2) &: \exists a, t[Happens\,(a, t) \wedge t_1 \leq t < t_2 \\ &\qquad \wedge Terminates\,(a, f, t)] \\ Declipped\,(t_1, f, t_2) &: \exists a, t[Happens\,(a, t) \wedge t_1 \leq t < t_2 \\ &\qquad \wedge Initiates\,(a, f, t)] \end{aligned}$$

Domain-independent component predicates:

$$\begin{aligned} HoldsAt\,(f, t_2) &: [Happens\,(a, t_1) \wedge Initiates\,(a, f, t_1) \\ &\qquad \wedge t_1 < t_2 \wedge \neg Clipped\,(t_1, f, t_2)] \\ \neg HoldsAt\,(f, t_2) &: [Happens\,(a, t_1) \wedge Terminates\,(a, f, t_1) \\ &\qquad \wedge t_1 < t_2 \wedge \neg Declipped\,(t_1, f, t_2)] \\ HoldsAt\,(f, t_2) &: [HoldsAt\,(f, , t_1) \wedge t_1 < t_2 \\ &\qquad \wedge \neg Clipped\,(t_1, f, t_2)] \\ \neg HoldsAt\,(f, t_2) &: [HoldsAt\,(f, t_1) \wedge t_1 < t_2 \\ &\qquad \wedge \neg Declipped\,(t_1, f, t_2)] \end{aligned}$$

We use the fluents and the above predicates to describe the fact that illustrate what is the effects of particular events triggered by the changes of the user location, application situation and resource status.

4.2. Service discovery policy scheme

In order to build a large scale distributed pervasive computing application, every entity should be self-descriptive and can define the policy acting on own one or the other entities. The context-aware service composition is a policy-driven composition process, where multiple services are connected via functional and data dependencies. In our scheme, service discovery is thought as a policy to enhance the system adaptation capability. When an application entity finds the conflict or conflict occurs between the entities, an appropriate entity is responsible for performing the specific policy. Some works focus on the access control policy [19], and we assume the service access is valid for all free service.

According to the proposed policy model and description method, we put forward a service discovery scheme by means of ontology. As shown in Fig. 5, the entity policy has the service discovery class that has five properties: *delegate*, *location*, *user*, *trigger condition* and *discover*. These properties are used to resolve the execution condition and action of service adaptation. Here, we describe the part of property set composing of the entity ontology. The role of a user determines the right of a specific user group, such as access control right or priority.

4.3. An example of scenario

To illustrate the usage of policy, consider a scenario of a mobile user running a multimedia streaming application on the PDA device. As Fig. 2(a) shown, s1 is a media source, and s2 is transcoder service node which can dynamically alter the frame rate, frame size, while s3 is mobile device. When mobile users enter the smart space, they want to use the device service in this room as media presentation device. If he leaves the room, the media streaming can be redirected to the mobile device. In addition, when battery power is low or the wireless bandwidth is low, media application can adapt to find the appropriate device service to improve the media characters such as larger frame size or higher fidelity. We give the policies of user, application, environment and resource of the scenario in Table 1.

The user requires that the proper display service should be provided when battle level is below 30% or when he wants to see the video in best video quality. The streaming application supports the maximum frame size and frame rate when the network bandwidth is above 400 k BPS. The environment adopts the First Come First Served policy in cases when users have equal rights. The network policy emphasizes that application based on the TCP link should be guaranteed when wireless bandwidth is below 100 k BPS, which can determine the priority of the application adaptation action when multiple applications exist. The proper action can be achieved under the condition of satisfying the policy consistency and QoS constraints.

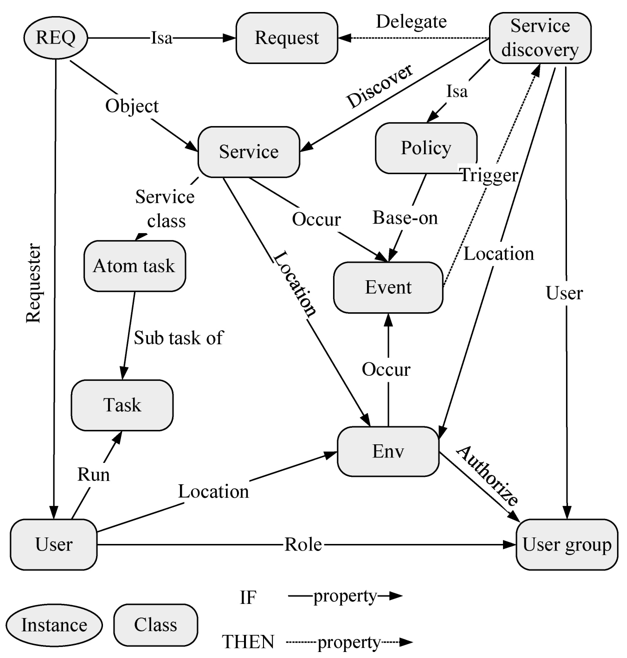

FIGURE 5. Service discovery policy scheme.

4.4. Proposed policy description

Our aim is to utilize multi-layer policy combination to realize spontaneous interoperation and application adaptation.

TABLE 1. A example of hierarchy policy description.

Layer	Conditions	Actions
User policies	Battle level below 30% best video quality, bandwidth is low	Discovery device service
Application policies	bandwidth <100 k, 400 k > bandwidth > 100 k, bandwidth > 400 k	Minimum/ Quantified/ Maximum frame size, frame rate
Environment policies	Equal right	FCFS (first come first served) Provide best-fit service
Resource policies	wireless bandwidth <100 k	TCP application preference

According to the scenario mentioned before, the policy can be defined as follows:

Events:

call_made if system call is performed
battery_low if battle level of mobile device $< 30\%$
in_room if user enter the room in the building
out_room if user walk out of the room
bandwidth_norm if bandwidth > 400 k
bandwidth_low if bandwidth < 100 k
wireless_bandwidth_low if wireless bandwidth < 100 k
presentation_low if presentation quality need to be improved
adaptation_accept if adaptation action is valid
composition_calls if composition action is performed

Fluents:

*Battery_low*_mode{
HoldsAt(*Battery_low_mode*, T_0);
¬*HoldsAt*(*Battery_low_mode*, T_0);
Initiates(*battery_low*, *Battery_low_mode*);
Terminate(*Battery_low_mode*);
}
Environment_room_mode{
HoldsAt(*Environment_room_mode*, T_0);
¬*HoldsAt*(*Environment_room_mode*, T_0);
Initiates(*in_room*, *Environment_room_mode*);
Terminate(*out_room*, *Environment_room_mode*)
}
Environment_floor_mode{
HoldsAt(*Environment_floor_mode*, T_0);
¬*HoldsAt*(*Environment_floor_mode*, T_0);
Initiates(*out_room*, *Environment_floor_mode*);
Terminate(*in_room*, *Environment_floor_mode*);
}
Basic_mode{
HoldsAt(*Basic_mode*, T_0);
¬*HoldsAt*(*Basic_mode*, T_0);
Initiates(*composition_calls*, *Transfer_mode*)
Terminate(*out_room*, *Transfer_mode*);
}
Transfer_mode{
HoldsAt(*Transfer_mode*, T_0);
¬*HoldsAt*(*Transfer_mode*, T_0);
Initiates(*composition_calls*, *Transfer_mode*)
Terminate(*out_room*, *Transfer_mode*);
}
Bandwidth_low_mode{
HoldsAt(*bandwidth_low*, T_0);
¬*HoldsAt*(*bandwidth_low*, T_0);
Initiates(*bandwidth_low*, *Bandwidth_low_mode*);
Terminate(*bandwidth_norm*, *Bandwidth_low_mode*);
}
Service_requirement_mode{
HoldsAt(*Service_requirement_mode*) : [
Happens(*do_discovery*, t_1)∧
Initiates(*do_discovery*, *Service_requirement_mode*, t_1)
$\wedge t_1 < t_2 \wedge \neg$*Clipped*($t_1$, *Service_requirement_mode*, t_2)
];
¬*HoldsAt*(*Service_requirement_mode*) : [
Happens(*do_discovery*, t_1)∧
Terminates(*do_composition*,
Service_requirement_mode, t_1)∧
¬*Declipped*(t_1, *Service_requirement_mode*, t_2)∧
$t_1 < t_2$];
}

Actions:

do_discovery
do_delegate
do_adaptation_assess
do_composition
do_end_composition
do_app_adaptation
do_adaptation_notify

Adaptation policy algorithm description:

(1) ^(battery_low | bandwidth_low | resentation_low) cause *do_discovery* if *Basic_mode* & (¬*Clipped*(*battery_low_mode*) | ¬*Clipped*(*bandwidth_low_mode*)) & (¬*Clipped* (*Service_requirement_mode*);
(2) *call_made* triggers *service_available if* (E.SS $<>$ *null*) & ($E.N_{\mathrm{cur}} < E.N_{\mathrm{MAX}}$) & ($R$.DR.$P_d >= Ui$.PU) & ($E$.SS.$SQ_j >= U_i$.SCU);
(3) *service_available* cause *do_adaptation_assess*;
(4) *adaption_accept* cause *do_adaptation_notify*;
(5) *in_room*cause *do_composition* if¬*Clipped*(*environment_room_mode*);
(6) *out_room* cause *do_end_composition*;
(7) ^(*bandwidth_low* | *battery_low*) cause *do_app_adaptation* if *emenvironment_floor_mode*;

The first policy defines the events and the conditions what can cause the service discovery action. The complex event triggering action occurs in an event set in which there are a sequence of instances of the event *battery_low*, *bandwidth_low* or *presentation_low*, and conditions satisfying three special fluents predicates. The second policy defines the selecting condition of the requisite service. The third policy defines the adaptation assess action to plan the service composition. The fourth and the seventh policies define propositions which drive action notification and only application adaptation strategy, respectively. When service composition action is valid, the notification of the action and the location of specified service should be sent to users. The only application adaptation action *do_app_adaptation* represents the action that reduces the media quality to satisfy end device requirement with out considering the enhancement of user experiences. The fifth and the sixth policy define control proposition of the composition action.

5. OPTIMIZATION ALGORITHM OF ADAPTATION POLICY

After Policy-driven service discovery adaptation action is triggered, how to select optimal service and adjust application action parameters to maximize the resource usage is the role of this section. This section describes the adaptation algorithm for adaptation service selection assessment. A new evaluation criterion is proposed.

5.1. Service selection requirement

Service selection is the core problem of the service discovery. We propose the policy description of service adaptation to make the service selection semantic-aware and QoS-aware.

Semantic-aware means the composition service can be changed in terms of the composite service semantic logic SL_i of the specific application. The new semantic logic SL_{ext} can satisfy the semantic consistency restriction: $SL_{\text{ext}} \Rightarrow SL_i$. We define the abstract ideas and actions as concepts and use different service composition plans to represent the concept of semantic fact. For example, the text presentation and the sound presentation of the corresponding text represent the same semantic concept. QoS-aware means that not only the input, output QoS parameters consistency between the elementary services, but the end-to-end QoS attributes, such as delay, jitter of application and environment requirement of QoS.

5.2. Adaptation optimization model

Considering the user and environment requirements, evaluation function is proposed as follows:

$$\alpha \sum_{i-\text{user}=1}^{N_{\text{user}}} \omega_{i-\text{user}} Q_{i-\text{user}} + \beta Q_{\text{env}}. \tag{1}$$

We introduce two quality functions of satisfaction: Q_{env} defines the quality of satisfaction of the specific environment in which the quality of satisfaction of *ith* user is denoted as $Q_{i-\text{user}}$. Where $w_{i-\text{user}}$ is the weight value of *ith* user relative to the specific environment, for example, in our smart classroom, the weight value of a teacher is higher than a student. α and β are coefficients that define the relative significance of Q_{env} and $Q_{i-\text{user}}$, and satisfy conditions: $\alpha + \beta = 1$. We regard them as a function of current user number N_{cur} of the environment. In other words, the adaptation aim is to maximize the associated quality of satisfaction of users and environments.

Next, the calculation of two kinds of quality of satisfaction can be defined as Equation (2) and Equation (3).

$$Q_{i-\text{user}} = \lambda \Delta P_{\text{quality}} + \Delta D_{i-\text{user}} - \Delta L_{i-\text{user}}, \tag{2}$$

$$Q_{\text{env}} = \Delta T + C + R_{\text{service}}, \tag{3}$$

where $\lambda = 1/P_{E-\text{quality}} - P_{R-\text{quality}}$ is a compensatary factor to guarantee most fit matching service selection for maximizing resource-using efficiency of the environment. In Equation (2) and Equation (3), we simply use the plus and minus to express the relation between the satisfaction degree and every sub-function only for formal requirement. Every sub-function can be regulated by the personal or environmental preference weight value. The weight-based normalization of the composite function is necessary to balance the influence of the several sub-functions. Other sub-functions are defined as follows:

$\Delta P_{\text{quality}} = P_{E-\text{quality}}/P_{R-\text{quality}}$ denotes the quality of satisfaction of the optional service which equals the ratio of current optional service quality and anticipant service quality. In my scenario, $\Delta P_{\text{quality}}$ is defined as the ratio of the screen size of two devices. Similar to the content value idea of Mohan and Smith [20], we define the Service quality value P_{quality} as follows:

$$P_{\text{quality}} = \sum_{i=1}^{n} \frac{\text{DR}_{ki} \text{ or } \text{SQ}_{ki} \text{ of optional service or device}}{\text{DR}_{ki} \text{ or } \text{SQ}_{ki} \text{ of user requirements}},$$

where i defines the *ith* quality of service attribute or device, n denotes the number of attributes.

$\Delta D_{i-\text{user}} = D_{R-i-\text{user}} - D_{E-i-\text{user}}$ denotes quality of improvement of delay time which equals to the difference between the user anticipation and estimating delay time which include the globe delay time change and local delay time change if service adaptation plan was executed. With respect to the calculation of delay in service composition and adaptation problem, it includes the service execute time, transmission time and composition time.

$\Delta L_{i-\text{user}} = f(t)$ denotes the difficulty degree of reaching the optional end service. According to physical space model PSM, the degree of difficulty can be transformed to function of time. We can adopt idea of the physical space location model proposed by Changhao Jiang [21] and get the parameter value according to the space tree structure of the physical environment and user mobility mode. This function realization will be further discussed.

Now

$$\Delta T = \sum_{i-\text{user}}^{N_{\text{user}}} T_{\text{candidate}} - \sum_{i-\text{user}}^{N_{\text{user}}} T_{\text{current}}$$

defines the change of total throughput. This function is for maximizing the wired/wireless network resource usage. This value can be determined by means of forecasting in terms of the application policy.

C denotes the capability of specific environment that is especially important for the physical environment, such as the number of users.

R_{service} denotes the rate of the service employment. In other words, it is defined as the ratio of employed service number to the number of services in the specific environments.

As mentioned before, we define the trigger condition of the service selection as $\Delta Q_{\text{satisfaction}} > 0$ which means the choice service can better improve the application and user experience. So the service selection is network optimization problem with limited conditions, which are user preference, application requirement and so on. Meanwhile, in the specific physical environment, the optional service is not exclusive but a service sets, so we propose the adaptation aim is Max $\{\Delta Q_0, \Delta Q_1, \Delta Q_2, \ldots\}$. This target that provides the most suitable, not the best, service selection is the result of balancing the environment and user requirements. On the other hand, this optimization process can figure out a priority service sequences as the candidate services, which support the service and user mobility.

5.3. Adaptation evaluation model detail discussion

Based on the adaptation evaluation model, the discussion of some important problems will be described in this section. The location model for physical space is proposed to support the location-aware service discovery and the service composition algorithm is given in detail. In the end, the performance analysis is elaborated.

5.3.1. The physical space model

The $\Delta L_{i-\text{user}} = f(t)$ expression and its computation is based on the hybrid location model. In this section, we will describe the hybrid location model in more detail and elaborate on how it can be used to express the semantic relation and compute the distance meant the difficulty degree of reaching the optional end service.

The appropriate physical space model is an important issue for the space service discovery and more broadly for context-aware application. Currently, the prevailing location models fall into two groups: the hierarchical symbol location model and the coordinate model. The hierarchical symbol model decomposes the physical environment in different levels of precision according to the coverage relation. The coordinate location model virtually grids the physical environment with a superimposed reference coordinate system which can be represented by a tuple of numbers. We adopt the hierarchical location model that is good for its implicit representation of spatial relationships, such as containment, closeness and supporting power of location query.

But most hierarchical location model cannot represent the connectivity relation, and cannot accurately explain the distance between two physical spaces. In our solution, we make use of the tree structure similar to B+ tree to model the physical space. The B+ tree is a version of B-tree(Balanced tree) that maintains a hierarchy of index and sequential linking of the data. This data structure can provide fast direct access and fast sequential access [22].

Fig. 6 illustrates part of the physical space model for FIT building of Tsinghua University. It is up to the architecture configuration of the building and social subjection relation. The tree style data structure for physical space model can efficiently handle queries of spatial relationship and semantic distance between different locations. In this figure, the parent-child link denoted by dash-dotted line means the super/sub space relation between spaces. The dash-dotted line with arrowhead represents not only the parent-child relation but the connectivity relation between the parent node and the child node. The same level link denoted by real line means the space connection relation.

The difference between our tree structure and traditional tree structure lies in the search method. The proposed tree structure defines the search from parent to child that it should be along the dash-dotted line link with arrow head, which guarantees the search process to be in accord with walk mode of people in physical space. At the same time, we define the different weight value for different links which lends itself to the computation of space semantic distance.

Search semantic relation. Our proposed physical space model embodies three searching semantic relations: coverage relation, adjacency relation and connection relation.

- Coverage relation: The parent-child link represents the semantic coverage relation about space location.
- Connection relation: The connection relation represents the space reachable based on the social administration

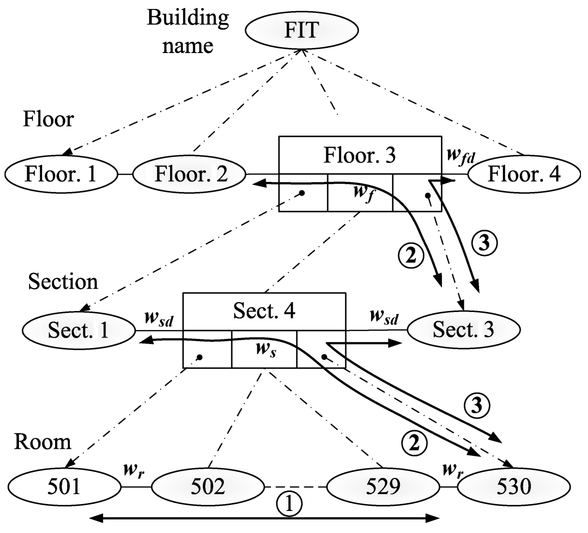

FIGURE 6. The physical space model.

domain and physical space. It includes the parent-child link represented by dash-dotted line with arrow head and same level link. The line (1) shows the connection relation between the same level nodes.

- Adjacency relation: The proposed tree data structure defines the adjacency relation between the child node and the sibling of parent based on the respective connection relation. The lines (2) and (3) show the searching based on the adjacency relation.

Semantic space distance. The proposed physical space model provide the efficient measure to compute the semantic space distance. Our service selection strategy can quantitatively get the cost of finding the appropriate service. As shown in Fig. 6, we define the different weight value according to the connectivity of physical space and ignore the weigh value of parent-child link. We propose that the semantic space distance between room r_1 and r_2 be a function of the three kinds of weigh value as follows:

$$D(r_1, r_2) = f(w_r, w_{sd}, w_{fd}),$$

where w_r is the walk cost between rooms in the same section, w_{sd} is the walk cost between sections in the same floor and w_{fd} is the walk cost between floors in the same building.

In addition, we introduce the two weigh value w_s and w_f in section and floor level to reflect space characters of the child node. The two weigh value can be regarded as the sum of weigh value in child level between the dash-dotted lines with arrow head. This two weigh values satisfy: $0 \leq w_s$, $w_f \leq \infty$. So the Semantic space distance equals to the minimum of the weigh value sum of searching path from room r_1 to r_2.

5.3.2. *Location-based service discovery*

As known to all, the primary goal of application for pervasive computing environments is to perform the user task given by user by exploiting the resources or services that are present in the neighborhood. The locality of service should satisfy the requirement that the reaching time between the service requesting entity and the service in composition should be as short as possible. Users prefer using the service or device within close proximities. We can add the location lag on the attribute of service or device to solve this problem.

In the process of service discovery, service request takes effect within specific physical zone with respect to the connectivity of physical environment and user location. User location can be attained from our location system; Cicada [23]. Our location-aware service discovery method add location-based weight value index on the top of service class discovery method similar GSD protocol [24]. The intelligent-forward service discovery strategy can effectively reduce the discovery space.

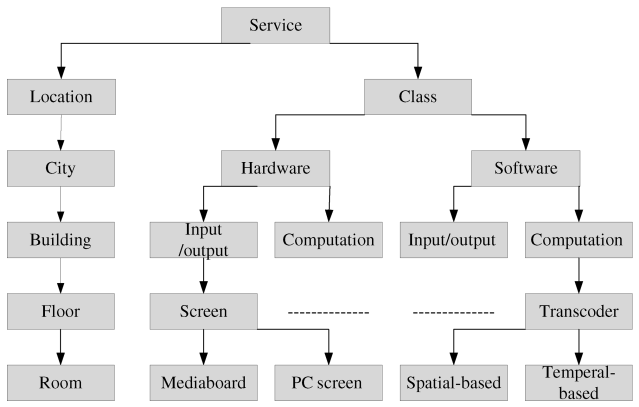

FIGURE 7. Hierarchical service attribute description.

Service description. In order to enhance the service interoperability, it is necessary to utilize the semantic service description to design service discovery method. Many semantic service descriptions use the OWL to define the service and resource ontology. Based on the attribute-value abstract of service ontology, we classify the service attribute as four sorts including service function class (SF), service location information (SL), service interface (SI), service parameter (SP). Our aim is to realize the multi-index service discovery method to provide intelligent discovery effect. In this paper, we only consider the service function class and service location information. As Fig. 7 shows, service function class represents the classification of semantic concept such as Screen and transcoder is service function class name.

Service advertisement. We consider application mode of pervasive computing as a peer-to-peer abstract and proposed content-based location solution for service discovery problem. Service advertisement will be controlled in limited hop numbers to keep the advertisement overhead low. Each service provider periodically advertises its service information and its neighborhood service function class with location weight value.

Service information summary. According to the advertisement information, each node constructs the local index to make service discovery more efficient. Here, we only emphasize the location-aware service index. Utilizing the physical space model, we can get weight value between two location points. A local node can organize the information as the service index item $\langle SF_i, \mathit{Location}, w_{li} \rangle$ in which SF_i is service function class of non-local node, *Location* is location information of SF node, and w_{li} is space semantic distance between local node and SF node. Owing to the location expression contain ordered relation, non-directly-neighborhood service index can be constructed by combination mode.

Combination mode. $\forall S_1, S_2, S_3$, S_1 is local service, S_2 is direct neighborhood of S_1, S_1 know S_3 via S_2, so

$$S_1 : \langle S_2, Location, w_{l2}\rangle, S_2 : \langle S_3, Location, w_{l3}\rangle$$

$$\begin{aligned}&if((S_1.Location \preceq S_2.Location\\ &\&S_2.Location \preceq S_3.Location)or\\ &\&(S_1.Location \succeq S_2.Location\\ &\&S_2.Location \succeq S_3.Location))\end{aligned}$$

$$\begin{aligned}&S_1 : \langle S_3, Location, w_{l2} + w_{l3}\rangle\\ &elseS_1 : \langle S_3, Location, |w_{l2} - w_{l3}|\rangle\end{aligned}$$

The $\preceq$ and the $\succeq$ represent the same-directional relation in the same traversing path of PSM. For example from Fig. 6:

$$\begin{aligned}&\text{FIT/Floor.3/Sect.4/503} \preceq \text{FIT/Floor.3/Sect.4/509}\\ &\text{FIT/Floor.3/Sect.1/509} \preceq \text{FIT/Floor.3/Sect.4/503}\end{aligned}$$

5.3.3. *Algorithm description*

According to the foundation of algorithm theory supporting adaptation evaluation model, we will give the detail of adaptation algorithm in this section.

The physical space model makes the terminal service selection location-aware. On obtaining terminal service set, the establishment of session path between the source and terminal node should satisfy the function requirement and QoS constraints.

When adaptation action is triggered, the user delegates the terminal service discovery requirement to local service node for location-aware service discovery. The location-aware service discovery algorithm is shown as Fig. 8.

On obtaining the valid terminal service set, we use adaptation composition algorithm to select optimal composition path for QoS-aware and semantic-aware. Fig. 9 shows the pseudo code of this algorithm.

The adaptation composition algorithm utilizes the service quality characters of terminal service node to select the middle service set for adaptation regulation. We assume that the previous session path is optimal, so based on the previous optimal path, the establishment of adaptation composition path is reasonable. The SCG is extended to tree-typed graph through the trim of edge among service nodes. The source node is root, terminal node is leaf node.

The trim of graph is to guarantee the validity of QoS parameters, such as bandwidth $\kappa_{\rm BW}$, service quality, which satisfy application session requirement. So the optimal path problem is transformed to multi-constraints optimal path problem of tree-type graph. We take account of the edge availability $E(v_{\rm l}, v_{\rm m})$. A and delay $E(v_{\rm l}, v_{\rm m})$. D of sub-path in SCG. By introducing the ranking value of sub-path $v_{\rm m}.r$ to calculate

```
Function Location-aware ServiceDiscovery ($N_c$, SQ)
{
    $N_d$ is initiator of service discovery, SQ is requisite service
        quality informtion.
    $N_c$ is current node for execute service discovery.
    $N_c$.List is neighbor node list with service information.
    Let $PL_i$ be the location of the i node in neighbor node list.
    w is distance bound for user preference.
    if Distance ($N_d$.Location, $N_c$.Location) < w
        if $N_c$.SQ satisfy SQ
            return $N_c$.
        else
            For all the neighbor service node
                calculate Distance ($N_c$.location, $PL_i$);
            Select k service nodes MinDisNode[k]
                with minimum distance to $N_c$;
            For every item of MinDisNode[k]
                Forword(MinDisNode[k], $N_c$, SQ).
}
```

FIGURE 8. Location-aware service discovery algorithm.

the optimal path from leaf node to source node, the algorithm returns the ranking list for composition path in increment order. The optimal path with minimal ranking value is selected, others is ready for session switch due to user mobility or new requirement.

5.3.4. *Performance analysis*

Our proposed location-aware service discovery method support two kinds of query: accurate location query and range query such as the closest display device. Service discovery request utilizes the local service index to semantically forward to nodes which most possibly have an answer.

A successful location-aware service discovery involves discovering available services and finally selecting a service proximate to user location of both network and physical space. We simulated the peer-to-peer location-aware service discovery algorithm using the popular network simulator ns-2. We define the basic location distance as 1. Every node has some neighbor nodes with long location distance in the random proportion.

We consider node topology distribution in the plane with 1000*1000 units. The initial location information is set up as follows: we group every 200*200 units to the same group with same location information, and location connectivity is shown as Fig. 10.

We evaluate the performance of the location-aware service discovery method by the metrics of the preciseness rate and the message load of discovery.

The preciseness rate. Most discovered services without considering the physical space location only have proximate network location such as network domain or hop number limit. But the service with most close network distance

```
Funtion AdaptationComposition (SCG)
{
    Session path satisfy SCG = (S, $L_c$, $L_d$), S = {$v_s$, $v_m$, $v_d$}.
    Terminal service node list: T_list = {$v_d$[j]}
    If middle service $S_b$ cannot regulate service quality for
    new terminal device service
        S_list = ServiceDiscovery($S_b$, $S_b$.SQ).
    For every item node $v_m$[i] in the S_list
        if E($v_s$, $v_m$[i]).BW ≥ $v_s.\kappa_{BW}$
            Insert(E($v_s$, $v_m$[i]), SCG)
            For every item node $v_d$[j] in the T_list
                if E($v_m$[i], $v_d$[j]).BW ≥ $v_m$[i].$\kappa_{BW}$
                    Insert(E($v_m$[i], $v_d$[j]), SCG)
        else delete $v_m$[i].
    Assume the leave nodes of SCG : $v_l$
    $v_l$.r = 0; $v_l.w_A$ =1; $v_l.w_D$ = 0;
    while $v_l$ is not $v_s$ do
        For every nodes $v_l$
        {   For every edge to partent node $v_m$
                {   $v_m.w_A = v_l.w_A$ * E($v_l$, $v_m$).A;
                    $v_m.w_D = v_l.w_D$ +E($v_l$, $v_m$).D;
                    $v_m$.r = $\delta v_m.w_D$ +$\theta$(1- $v_m.w_A$);
                    $v_m$.Q_list.insert($v_l$, $v_m$.r);
                }
            $v_l$ = $v_m$;
        }
    return vs.Q_list with increment order.
}
```

FIGURE 9. Adaptation composition algorithm.

maybe is not most available service in view of physical space constraint. Our method fully considers the physical distance between user and requisite services, and efficiently enhances the consistency between user requirement and discovery result. Let N_{hop} = the number of visited nodes satisfying location bound w through hop-based service discovery. Let N_{location} = the number of visited nodes satisfying location bound w through location-aware service discovery. We introduce the metric η to express the comparison of the discovery preciseness rate. The metric η is computed as follows:

$$\eta = \frac{N_{\text{location}}}{N_{\text{hop}}}$$

Based on the idea of GSD protocol [23], our method takes account of location preference, which keeps the GSD discovery efficiency and furthermore reduces the discovery message load by introducing the location-aware forward. With equal average advertisement frequency, GSD protocol forward the discovery request to nodes with relative service class information, but we select k nodes with minimal location weight value which is subsets of GSD forward node. Fig. 11 shows the location-aware service discovery method has higher precision rate for location-related service discovery, correspondingly it has lower message load.

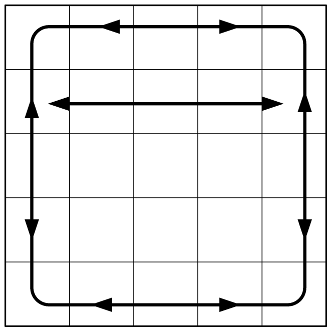

FIGURE 10. Location connectivity in simulation.

Composition time. As Fig. 12 shown, our adaptation composition algorithm is two stage process, one is edge trim from node V_s to node V_l, another is ranking of path from V_l to V_s.

Let e_i = the number of edge from every node V_m [i] to V_l in extended SCG. Let n_m = the number of node V_m in extended SCG. So, the algorithm has a runtime of O $(\sum_{i=1}^{n_m} e_i + n_m)$. In pervasive computing environment, user and service mobility is an important aspect. This aspect can change the composed solution and trigger new the adaptation requirement. We can utilize service discovery to find k nearest neighbor services first. The composition algorithm of adaptation can get an available service path with ordered priority about quality of satisfaction. When user or service mobility occurs, it is possible to ensure the appropriate service can be attained with minimal re-composition time from the result of adaptation evaluation last time.

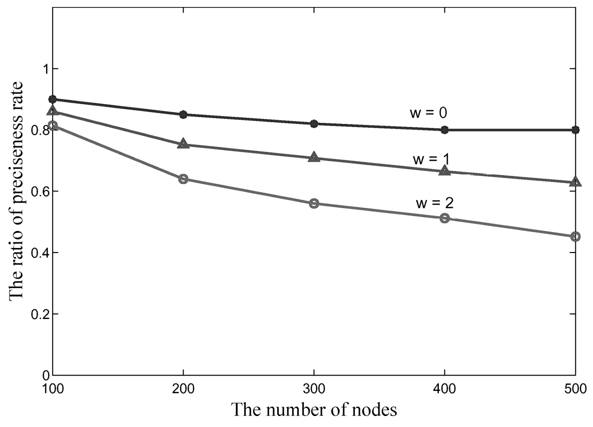

FIGURE 11. Comparison of precision rate.

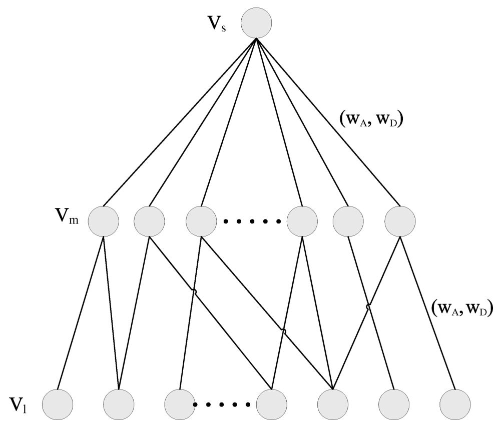

FIGURE 12. Service composition path selection.

6. CONCLUSION AND FUTURE WORK

In this paper, we present a policy-driven service discovery scheme for adaptation of service composition application especially multimedia application in pervasive computing environments. The adaptation approach uses the policy to get high-level goal and management strategy of the relative entity, and forms adaptation action. The proposed evaluation algorithm efficiently carries out optimization of the service selection supporting location-aware. Our discussion demonstrates that the method improves the service selection preciseness rate as well as network load and reduced the process time of re-composition resulting form user or service mobility.

We will further study multi-policy automatic combination and reasoning problems, design the multi-index-based service discovery method in detail, and implement our prototype system.

FUNDING

This work was supported by Program for New Century Excellent Talents in UniversityNCET-04-0079 and Specialized Research Fund for the Doctorial Program of Higher Education, No. 20050003048.

REFERENCES

[1] NIST-Smart Space, http://www.nist.gov/smartspace

[2] Satyanarayanan, M. (2001) Pervasive computing vision and challenges. *IEEE Pers. Comm.*, **6**, 10–17.

[3] Prithwish, B., Wang, K. and Thomas, D.C. (2003) Dynamic task-based anycasting in mobile ad hoc networks. *ACM/ Kluwer J. Mobile Netw. Appl.*, **8**, 593–612.

[4] Kindberg, T. and Fox, A. (2002) System software for ubiquitous computing. *IEEE Pervasive Computing*, **1**, 70–81.

[5] Nancy, S. and Karmouch, A. (2005) An automated policy-based management framework for differentiated communication systems. *IEEE J. Select. Areas Commun.*, **23**, 2236–2247.

[6] Xiaohui, G. http://cairo.cs.uiuc.edu/publications/paper-files/ thesis_xiaohui.pdf

[7] Zeng, L.Z., Benatallah, B., Ngu, A.H.H., Dumas, M., Kalagnanam, J. and Chang, H. (2004) QoS-aware middleware for web services composition. *IEEE Trans. Softw. Eng.*, **30**, 311–327.

[8] Liao, C.Y., Shi, Y.C. and Xu, G.Y. (2000) AMTM – An Adaptive Multimedia Transport Model. *Proc. of SPIE Int. Symp. Voice, Video, and Data Communication*, Boston, MA, USA, 5–8 November, 141–149. SPIE, Bellingham, WA, USA.

[9] Gu, X.H. and Nahrstedt, K. (2006) Distributed multimedia service composition with statistical QoS assurances. *IEEE Trans. Multimedia*, **8**, 141–151.

[10] Paterno, F., Mancini, C. and Meniconi, S. (1997) ConcurTaskTrees: A Diagrammatic Notation for Specifying Task Models. *Proc. IFIP TC13 Int. conf. Human–Computer Interaction*, Sydeny, Australia, 14–18 July, pp. 362–369. Chapman & Hall, London, UK.

[11] Jin, J.W. and Nahrstedt, K. (2004) QoS specification languages for distributed multimedia applications: a survey and taxonomy. *IEEE Multimedia*, **11**, 74–87.

[12] Satyanarayanan, M. (2004) From the editor in chief: the many faces of adaptation. *IEEE Pervasive Computing*, **3**, 4–5.

[13] Lymberopoulos, L., Lupu, E. and Sloman, M. (2002) An Adaptive Policy Based Management Framework for Differentiated Services Networks. *Proc. Policies for Distributed Systems and Networks*, Monterey, California, USA, 5–7 June, pp. 147–158. IEEE, Los Alamitos, California.

[14] Damianou, N., Dulay, N., Lupu, E.C. and Sloman, M.S. (2001) The Ponder Policy Specification Language. *Proc. Workshop on Policies for Distributed Systems and Networks*, 29–31 January, pp. 18–38. Springer, Berlin/Heidelberg.

[15] Anderson, A. (2004) An Introduction to the Web Services Policy Language (WSPL). *Proc. Fifth IEEE Int. Workshop on Policies for Distributed Systems and Networks*, New York, USA, 7–9 June, pp. 189–192. IEEE, Washington, DC, USA.

[16] Kagal, L., Finin, T. and Anupam, J. (2003) A policy language for a pervasive computing environment. *Proc. 4th Int. Workshop on Policies for Distributed Systems and Networks*, 4–6 June, pp. 63–74. IEEE, Los Almitos, California.

[17] Lobo, J., Bhatia, R. and Naqvi, S. (1999) A Policy Description Language. *Proc. Sixteenth National Conf. Artificial Intelligence and Eleventh Innovative Applications of Artificial Intelligence*, Orlando, Florida, USA, 18–22 July, pp. 291–298. AAAI/ MIT, Menlo Park, USA.

[18] Kowalsky, R. (1986) A logic-based calculus of events. *New Generat. Comput.*, **4**, 67–95.

[19] Syukur, E., Loke, S.W. and Stanski, P. (2005) Methods for policy conflict detection and resolution in pervasive computing environments. *Proc. Policy Management for Web workshop in conjunction with WWW 2005 Conf.*, Chiba, Japan, 10–14 May, pp. 13–20. ACM, Danvers, MA, USA.

[20] Mohan, R., Smith, J.R. and Li, C.S. (1999) Adapting multimedia internet content for universal access. *IEEE Trans. Multimedia*, **1**, 104–114.

[21] Jiang, C.H. and Steenkiste, P. (2002) A hybrid location model with computable location identifier for ubiquitous computting. *Proc. 4th Int. Conf. Ubiquitous Computing*, 29 September-1 October, pp. 246–263, Springer, London, UK.

[22] Thomas, H.C., Charles, E.L., Ronald, L.R. and Clifford, S. (2001) B-Trees. In Bob, P. and Betsy, J. (eds), *Introduction to Algorithms* (2nd edn). MIT Press and McGraw-Hill, Cambridge, Massachusetts London, England.

[23] Gu, H.L., Shi, Y.C., Chen, Y., Wang, B.B. and Jiang, W.F. (2006) Cicada: a highly-precise, easy-embedded and omnidirectional indoor location sensing system. *Proc. 1st Int. Conf. Grid and Pervasive Computing*, 3–5 May, pp. 385–394, Springer, Berlin/Heidel-berg.

[24] Chakraborty, D., Joshi, A., Yesha, Y. and Finin, T. (2006) Toward distributed service discovery in perva sive computing environments. *IEEE Trans. Mobile Comput.*, **5**, 97–112.